语法制导翻译

2020年7月2日 22:46

1. 语法制导翻译(Syntax-Directed Translation)
    a. 语法制导翻译：语法分析 和 语义翻译 同时实现
        i. 语义翻译：语义分析 和 中间代码生成 同时实现
        ii. 即语法制导翻译包含：**语法分析、语义分析、中间代码生成**
        iii. 语法制导翻译使用CFG引导对语言的翻译，是一种面向文法的翻译技术
    b. 基本思想
        i. 如何表示语义信息：为CFG的文法符号设置语义属性
        ii. 如何计算语义属性：用与文法符号所在产生式 （语法规则）相关联的**语义规则**来计算文法符号的语义属性
            1）对于给定的输入串x，构建x的语法分析树，并利用与产生式（语法规则）相关联的语义规则来计算分析树中各结点对应的语义属性值
        iii. 如何将语义规则和语法规则（产生式）关联：语法制导定义和语法制导翻译方案
    c. 语法制导定义(Syntax-Directed Definitions, SDD)
        i. 是对CFG的推广，将**每个文法符号和一个语义属性集合相关联**
        ii. 将每个产生式和一组语义规则相关联，这些规则用于计算该产生式中各文法符号的属性值
        iii. 如果X是一个文法符号，a是X的一个属性，则用X.a表示属性a在某个标号为X的分析树结点上的值
    d. 语法制导翻译方案(Syntax-Directed Translation Scheme , SDT)
        i. **在产生式右部嵌入了程序片段（语义动作）**的CFG
        ii. 按照惯例，语义动作放在花括号内
        iii. 一个语义动作在产生式中的位置（花括号的位置）决定了这个动作的执行时间
    e. SDD和SDT的关系
        i. SDD：是关于语言翻译的高层次规格说明；隐蔽了许多具体实现细节，使用户不必显式地说明翻译发生的顺序
        ii. SDT：可以看作是对SDD的一种补充，是SDD的具体实施方案；显式地指明了语义规则的计算顺序，以便说明某些实现细节
2. 语法制导定义SDD
    a. 文法符号的属性：综合属性 和 继承属性
    b. 综合属性(synthesized attribute)：在分析树结点N上的非终结符A的综合属性只能通过N的子结点或N本身的属性值来定义
        i. 终结符可以具有综合属性。终结符的综合属性值是由词法分析器提供的词法值，因此在SDD中没有计算终结符属性值的语义规则
        ii. 如加法表达式的值属性

c. 继承属性(inherited attribute)：在分析树结点N上的非终结符A的继承属性只能通过N的父结点、N的兄弟结点或N本身的属性值来定义
    i. 终结符没有继承属性。终结符从词法分析器处获得的属性值被归为综合属性值
    ii. 如定义变量表达式的类型属性
d. 副作用(side effect)：语义规则用于定义产生式左部文法符号综合属性的规则
    i. 一般是额外调用一个函数过程
e. 注释分析树(Annotated parse tree)：每个结点都有属性值的分析树
f. 属性文法(Attribute Grammar)：没有副作用的SDD
    i. 属性文法的规则仅通过其他属性值和常量来定义属性值

3. SDD的求值顺序
    a. 语义规则建立了属性之间的依赖关系，在对语法分析树节点的一个属性求值之前，必须首先求出这个属性值所依赖的所有属性值
    b. 依赖图(Dependency Graph)
        i. 是一个描述了分析树中结点属性间依赖关系的有向图
        ii. 分析树中每个标号为X的结点的每个属性a都对应 着依赖图中的一个结点
        iii. 如果属性X.a的值依赖于属性Y.b的值，则依赖图中有一条从Y.b的结点指向X.a的结点的有向边
        iv. 有时需要建虚结点来接受从子结点属性和本身属性共同决定的属性
    c. 属性值的计算顺序
        i. 可行的求值顺序是满足下列条件的结点序列N1, N2, … , Nk：如果依赖图中有一条从结点Ni到Nj的边(Ni→Nj)，那么i < j（即：在节点序列中，Ni 排在 Nj 前面）
        ii. 这样的排序将一个有向图变成了一个线性排序，这个排序称为这个图的拓扑排序(topological sort)
        iii. 对于只具有综合属性的SDD，可以按照任何自底向上顺序计算值（必无环，必有拓扑排序）
        iv. 对于同时具有继承属性和综合属性的SDD，不能保证存在一个顺序对各个节点上的属性进行求值（因为可能有环，而无拓扑排序）
4. S-属性定义和L-属性定义
    a. 作用
        i. 从计算的角度看，给定一个SDD，很难确定是否存在某棵语法分析树，使得SDD的属性之间存在循环依赖关系
        ii. 幸运的是，存在一个SDD的有用子类，它们能够保证对每棵语法分析树都存在一个求值顺序，因为它们不允许产生带有环的依赖图
        iii. 不仅如此，S-SDD和L-SDD可以和自顶向下及自底向上的语法分析过程一起高效地实现
    b. S-属性定义(S-Attributed Definitions, S-SDD)：仅仅使用综合属性的SDD称为S属性的SDD，或S-属性定义、 S-SDD
        i. 如果一个SDD是S属性的，可以按照语法分析树节点的任何自底向上顺序来

计算它的各个属性值

ii. S-属性定义可以在自底向上的语法分析过程中实现

c. L-属性定义(L-Attributed Definitions, L-SDD)：在一个产生式所关联的各属性之间，依赖图的边可以从左到右，但不能从右到左

i. 充要条件：当且仅当它的每个属性要么是一个综合属性，要么是满足如下条件的继承属性：假设存在一个产生式A→X1X2...Xn，其右部符号Xi (1<=i<=n)的继承属性仅依赖于下列属性：

1) A的继承属性

2) 产生式中Xi左边的符号X1, X2, ... , Xi-1的属性

3) Xi本身的属性，但Xi 的全部属性不能在依赖图中形成环路

ii. 充分条件：每个S-属性定义都是L-属性定义

5. 语法制导翻译方案SDT

a. S-SDD转换SDT：将每个语义动作放在产生式最后

i. 即当**归约发生时执行相应的语义动作**

ii. 如果一个S-SDD的基本文法可以使用LR分析技术，那么它的SDT可以在LR语法分析过程中实现

iii. 扩展的LR语法分析栈：在栈中使用附加域存放综合属性。注意综合属性可能有多个

iv. 将语义动作中的抽象定义式改写成具体的栈操作。形如 stack[top-2].value=f(stack[top].value,stack[topp-2].value); top-=2;

b. L-SDD转换为SDT：

i. 将计算某个非终结符号A的继承属性的动作插入到产生式右部中紧靠在A的本次出现之前的位置上

ii. 将计算产生式左部符号的综合属性的动作放在这个产生式右部的最右端

iii. 如果一个L-SDD的基本文法可以使用LL分析技术（SELECT互不相交），那么它的SDT可以在LL或LR语法分析过程中实现

6. 在非递归的预测分析过程中进行翻译

a. 扩展后的语法分析栈：指向将被执行的语义动作代码的指针；A的继承属性；A的综合属性

b. 分析栈中的每一个记录都对应着一段执行代码

i. 综合记录出栈时，要将综合属性值复制给后面特定的语义动作

ii. 变量展开时（即变量本身的记录出栈时），如果其含有继承属性，则要将继承属性值复制给后面特定的语义动作

7. 在递归的预测分析过程中进行翻译

a. 为每个非终结符A构造一个函数，A的每个继承属性对应该函数的一个形参，函数的返回值是A的综合属性值。形如A_syn A(token, A_inh)

b. 对出现在A产生式右部中的每个文法符号的每个属性都设置一个局部变量

c. 对于每个语义动作，将其代码复制到语法分析器，把对属性的引用改为对相应变量的引用

8. L-属性定义的自底向上翻译

a. 以LL文法为基础的L-SDD，可以通过修改文法，在LR语法分析过程中计算这个新文法之上的SDD
    i. 首先构造SDT，在各个非终结符之前放置语义动作来计算它的继承属性，并在产生式后端放置语义动作计算综合属性
    ii. 将所有的语义动作替换成新引入的标记非终结符(Marker Nonterminal)
        1) 标记非终结符只产生空串，语义动作在该产生式的末尾，使得修改后的SDT的所有语义动作都出现在产生式末尾
    iii. 设形如A→α{a}β的产生式中替换了语义动作a为标记非终结符M
        1) 将动作a需要的A或α中符号的任何属性作为M的继承属性进行复制
        2) 按照a中的方法计算各个属性，将计算得到的这些属性作为M的综合属性